Plataforma IoT per a aplicacions LoRaWAN

Roger Gili i Coscojuela

17/06/2021

Resum– L'objectiu del TFG és crear una plataforma IoT de codi obert per a sensors que utilitzin una xarxa LPWAN, en concret, utilitzant una d'aquestes tecnologies anomenada Long Range Wireless Area Network (LoRaWAN). El projecte inclou la creació d'un firmware per als nodes i la integració dels mateixos a la xarxa LoRaWAN, la configuració del middleware per recollir les dades i l'enviament de les mateixes a un backend per a tractar-les i guardar-les. Per últim també s'ha creat un frontend de visualització de dades, intentant que sigui el màxim independent del proveïdor de la infraestructura escollida, tant de la de comunicacions i edge com de la de cloud.

Paraules clau– Plataforma IOT, LoRaWan, LPWAN, STM32

Abstract– The main goal of the TFG is to create an open source IoT platform for sensors that use a LPWAN network, specifically using one of these technologies called Long Range Wireless Area Network (LoRaWAN). The project includes the creation of a firmware for the nodes and their integration in the LoRaWAN network, the configuration of the middleware to collect and send the data to a backend and process and save it to a database. Finally, a data visualization frontend has also been created, with the aim to be as independent as possible from the infrastructure provider chosen, both for communications, edge and cloud.

Keywords– IOT Platform, LoRaWan, LPWAN, STM32

1 Introducció

DURANT els últims anys s'ha experimentat un augment exponencial de l'anomenat Internet of Things (IOT), que és la interconnexió digital d'objectes quotidians amb internet. Dins aquest auge, han aparegut o s'han popularitzat noves tecnologies d'interconnexió per a aquests dispositius anomenades Low-power wide area networks (LPWANs). Aquestes noves tecnologies s'enfoquen en millorar algun tipus de compromís entre els següents aspectes: rang de transmissió, amplada de banda, consum energètic o cicle de treball.[1]

En el cas dels dispositius IoT, una de les seves característiques habituals és la poca informació que es necessita enviar (sensors) o rebre (actuadors) des del o al node corresponent. El responsable de gran part del consum energètic d'aquests dispositius normalment radica en l'enviament de dades, per tant, reduir la quantitat de dades i la seva periodicitat (cicle de treball) a enviar es tradueix en millores considerables en la vida útil de la bateria d'aquests aparells, associada a l'ús de diferents modes d'estalvi energètic programables per SW. Entre les tecnologies de transmissió de dades que s'empren en aquest tipus de dispositius destaquen, en el cas de transmissió de dades d'abast curt, Bluetooth i ZigBee entre d'altres, mentre per llargues distàncies les més populars actualment són SigFOX, NB-IoT o LoRa[2]

Aquests dispositius (nodes) es connecten a servidors (núvol) a través d'una infraestructura de gateways (edge) donant lloc a la cadena IoT. Aquest paradigma ha motivat també l'aparició de plataformes, com per exemple MyDevices[3], ThingsBoard[4] o Thinker[5], la finalitat de les quals és gestionar les dades rebudes d'aquests dispositius IoT.

2 Objectius

Tot i la quantitat de virtuts que ofereix LoRaWAN, les solucions que utilitzen aquesta tecnologia solen ser propietàries i de codi tancat. Per aquest motiu, l'objectiu d'aquest projecte és contribuir a construir una plataforma IoT de codi obert per a sensors i actuadors connectats a una xarxa LoRaWAN.

Aquest projecte s'executa en paral·lel a una plataforma HW per al device en un TFM d'Enginyeria de Telecomu-

• E-mail de contacte: roger.gili@e-campus.uab.cat

• Menció realitzada: Enginyeria de Computadors

• Treball tutoritzat per: Jordi Carrabina Bordoll (Departament de Microelectrònica i Sistemes Electrònics) i Marc Codina Barberà (Departament de Microelectrònica i Sistemes Electrònics)

• Curs 2020/21

nicacions. Aquest paper se centra en l'apartat software i firmware de la plataforma IoT, que inclou la creació d'un firmware per als nodes, la integració dels mateixos a una xarxa LoRa, la configuració del middleware per recollir les dades i l'enviament de les mateixes a un backend per a tractar-les i guardar-les. Per últim, la creació d'un frontend de visualització, que sigui el màxim independent del proveïdor de la infraestructura, tant de la de comunicació i edge com de la de cloud, aspectes que conformen els subobjectius del projecte.

El punt d'enllaç amb el HW dels nodes, és el microcontrolador STM32WL sobre el qual es construirà el firmware. Aquest processador és el primer que integra al SoC una MCU ULP amb un mòdul RF subGHz, adaptable a diferents protocols. El SoC permet una connexió flexible a sensors amb protocols estàndard (I2C, SPI, GPIO, USART). Com a xarxa principal seleccionem la xarxa LoRaWAN de The Things Network[10], de gran abans mundial i construïda a partir de comunitats d'usuaris. La flexibilitat i interoperabilitat es validarà sobre la xarxa d'Everynet [11]. A més, alguns nodes també podran tenir actuadors, per tant, hauran de ser capaços de rebre també peticions a través de la xarxa per tal d'activar-los.

El backend serà l'encarregat de rebre la informació de les xarxes LoRa esmentades, processar les dades rebudes i guardar-les en una base de dades. Aquest backend també serà el responsable de generar una resposta als nodes en cas que sigui necessari, per exemple, quan la recepció d'una dada activi un dels disparadors.

Per últim, el frontend permetrà als usuaris poder veure de forma gràfica les dades rebudes pels sensors, així com afegir nous nodes o disparadors al sistema. El sistema també avisarà a l'usuari de quant un dispositiu no ha enviat dades durant un temps determinat, la qual cosa permetria detectar per exemple un dispositiu espatllat o que s'ha quedat sense bateria.

3 METODOLOGIA I PLANIFICACIÓ

La metodologia emprada en el projecte és un model de waterfall. Tot i això, el desenvolupament de les diferents parts del projecte - device, comunicacions/edge, frontend/backend cloud - s'ha realitzat sobre una metodologia incremental basada en iteracions, que ha permès anar implementant de forma gradual els requisits de la plataforma a la vegada que hem pogut mostrar l'avançament del projecte amb versions usables d'aquest.

Per tal de planificar els desenvolupaments i assegurar el compliment dels objectius, s'ha desenvolupat un diagrama de Gantt amb les diferents fites, el qual ha funcionat com a eina de control per tal de demostrar els avenços realitzats, mantenint a la mateixa vegada les dates planificades per a cada fase del projecte. El diagrama de Gantt es pot consultar a l'annex d'aquest mateix document. Així mateix, el codi desenvolupat, tant del device com del backend i frontend es pot trobar al github del projecte.

Com a eines per a desenvolupar el projecte, he utilitzat: Per a la programació i debug de la placa l'IDE de STM CubeMX , ja que permet autogenerat la majoria de codi inicial de forma gràfica.

Per al disseny de la base de dades Mysql Workbench Developer, amb la que s'han generat l'esquema que es mostra a l'annex, i permet generar a partir d'aquests esquemes els scripts per a crear les taules necessàries. Per tal de poder visualitzar les dades que s'anaven guardant a la mateixa he utilitzat phpMyAdmin.

Per últim, el disseny de la web està basat en el framework de Bootstrap. S'han utilitzat les llibreries externes de jQuery i Chart.js per a la visualització de les dades de forma gràfica.

4 ARQUITECTURA DEL SISTEMA

A la figura 1 podem veure l'arquitectura típica sobre la qual es modelen la majoria de sistemes IoT. Els devices, també anomenats algunes vegades nodes, són dispositius equipats amb sensors i/o actuadors, amb uns recursos limitats tant de còmput com d'emmagatzematge, i normalment amb bateries de dimensions reduïdes que ha de durar llargs períodes de temps (p. ex. deu anys en el cas dels comptadors domèstics d'aigua o gas). Aquests dispositius són els responsables de recollir les dades - comptador de passos, freqüència cardíaca, lluminositat, polsós del comptador, etc.- i transmetre-les al cloud a través del gateway/edge.

L'edge, per la seva part, està compost pels dispositius que s'encarreguen de recollir les dades de diferents nodes sensors i retransmetre-les al cloud. Hi ha dispositius edge que poden tractar les dades abans d'enviar-les al cloud, com per exemple els casos dels mòbils amb els wearables o dels home gateways points en domòtica, de manera que la reacció sobre els actuadors és més ràpida. En el cas de les xarxes LPWAN, no és la seva principal funció.

Per últim, el cloud és l'encarregat d'emmagatzemar i tractar les dades rebudes. Al cloud es troba normalment l'aplicació final amb la qual l'usuari interactua.

Una aplicació típica d'aquesta arquitectura la trobem per exemple en els wearables amb podòmetre actual. El braçalet (device) envia les dades del podòmetre al mòbil que fa d'interfície d'usuari, que les tramet al cloud on són emmagatzemades i processades i, potencialment, compartides (p. ex. runners). Finalment, l'usuari pot, mitjançant el seu mòbil, consultar també les dades del cloud a través d'una aplicació (end-user app).

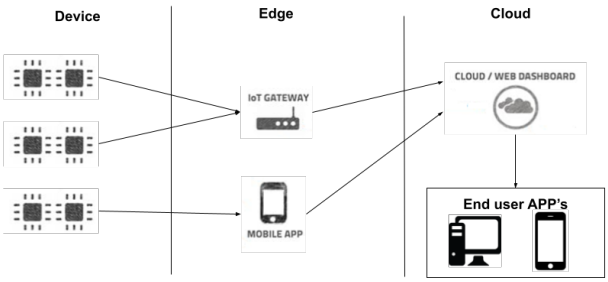


Fig. 1: Arquitectura típica IOT

4.1 Device

El device està basat en el SoC STM32WL55JC. La particularitat d'aquest SoC és que incorpora en el mateix encapsulat un processador Arm® Cortex®-M4/M0, a més d'un transceptor amb suport LoRa. El fet d'integrar en el mateix

encapsulat processador i transceptor permet de fer dissenys més petits i en general consumeix menys energia.

Mentre es desenvolupa la plataforma HW en paral·lel, el meu desenvolupament es basa en una plataforma de prototipat NUCLEO-WL55JC que conté el mateix processador juntament amb l'antena i una interfase bàsica per a connectar-hi sensors i actuadors; així com per connectar-la a un PC i descarregar-hi codi des de l'entorn de desenvolupament.

A la figura 2 podem observar que el fil principal del programa processa els esdeveniments que es troba a la cua d'esdeveniments (implementats en un registre de 32 bits). Quan acaba de processar-los tots, entra en un mode de baix consum (sleep).

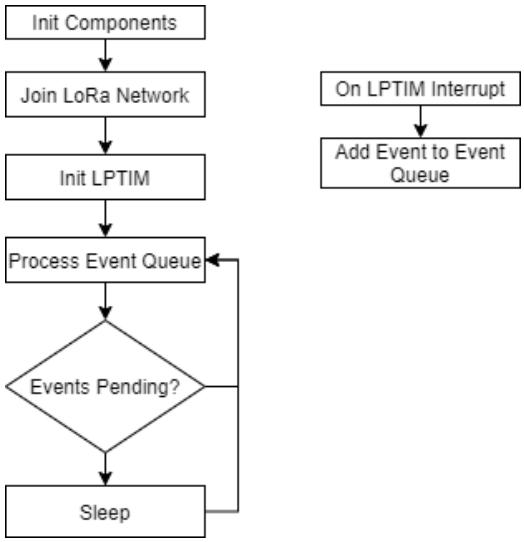


Fig. 2: Diagrama de flux del device

El Low Power Timer, per altra banda, genera una interrupció associada al cicle de treball (cada cop que han passat els segons o minuts especificats). La rutina d'interrupció del LPTIM s'encarrega d'indicar que s'ha de realitzar una nova lectura dels sensors i enviar-la a través de la interfície LoRa. La manera d'indicar que s'ha de realitzar aquesta acció és modificar el bit corresponent en el registre que gestiona la cua d'esdeveniments.

La cua d'esdeveniments, com ja s'ha indicat, es gestiona internament amb un registre de 32 bits (per la qual cosa podem tenir un màxim de 32 accions dins del nostre programa). L'encarregat de realitzar les accions d'aquesta cua és el seqüenciador[6], que és una funció que executa les funcions del programa al qual li hem associat a cadascun dels bits del registre d'esdeveniments, amb la particularitat que cadascuna de les funcions s'executa en una secció crítica.

Les dades llegides pels sensors són enviades a través de la xarxa de radiofreqüència LoRaWan utilitzant el format de dades especificat en la secció 4.1.2.

El backend de la nostra plataforma pot enviar dades als nostres devices, mitjançant un missatge de downlink. Quan un device rep un d'aquests missatges, descodifica el seu payload per tal de saber sobre quin actuador ha de treballar [Figura 3].

```
###### ========== TxData ============

###### ========== Reply from AS ==========
###### D/L FRAME:0000 | SLOT:2 | PORT:2 | DR:3 | RSSI:0 | SNR:11

Received buffer size 3

Aplication requested to put actuator 1 to 1 value
```

Fig. 3: Exemple de com el device reacciona a un missatge de downlink, vist a través de la UART

4.1.1 Communicacions - Protocol LPWAN LoRaWAN

La xarxa LPWAN escollida, anomenada LoRaWAN, està específicament dissenyada per a aplicacions IoT, amb l'objectiu de poder connectar els diferents nodes (d'un usuari) a un servidor centralitzat (indicat pel mateix usuari).

El procés de normalització del protocol LoRa el fa la LoRa Alliance[7]. La capa física està basada en la tècnica coneguda amb el nom de Chirp spread spectrum per a la transmissió de dades, i utilitza les freqüències ISM lliures (sense llicència) de ràdio SUB-Ghz: 868 Mhz [i 433 Mhz] (Europa), 915 Mhz (Austràlia i Nord Amèrica) 867 Mhz (Índia) i 923 Mhz (Asia). La implementació de la capa Media Access Control (MAC) dóna lloc a la xarxa LoRaWAN, que és la que hem emprat en la nostra plataforma [Figura 4].

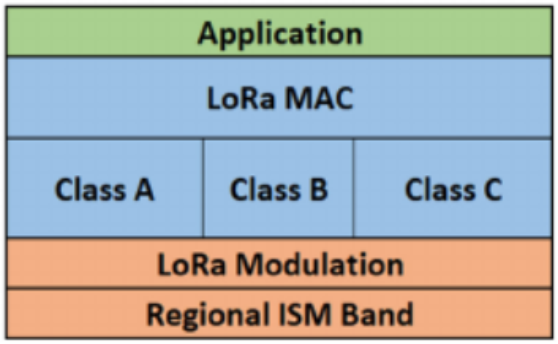


Fig. 4: Vista general de la pila del protocol LoRa

LoRaWAN pot aconseguir transmetre dades a una distància entre 2 i 5 Km en àrees urbanes i fins a 15 km en àrees suburbanes[8][9]. En aquest aspecte, LoRa permet una amplada de banda variable/adaptativa, que permet sacrificar bitrate en favor d'una major distància coberta. L'elecció d'una major o menor distancia ve determinada per l'spread factor utilitzat, la qual cosa limita també la velocitat de l'enllaç.

Per últim, a causa de limitacions legals d'utilització sobre l'espectre de radiofreqüència utilitzada, la quantitat màxima de dades que podem enviar en un paquet ve també determinada pel SF i la data rate utilitzat (cicle de treball). Tot i això, es considera que és una transmissió "bruta"que està sent criticada en diferents països (Holanda, Suïssa) per la contaminació RF que provoca.

Per tal de poder unir-se a la xarxa, cada dispositiu necessita una adreça de xarxa (DevAddr) i unes claus de sessió per a poder-se unir a la xarxa LoRaWAN. Aquest DevAddr està compòs per una NwkAddr (adreça del dispositiu dins la xarxa) unida a un NwkID (identificador de xarxa). Exis-

teixen dues maneres diferents que permeten activar un nou dispositiu a la xarxa, i rebre d'aquesta manera el DevAddr i les claus de sessió, necessàries per a identificar un dispositiu de forma única dins la xarxa. Per una banda tenim l'ABP (Activation By Personalization) i per altra l'OTAA (Over-The-Air Activation).

En l'activació a través de l'aire (OTAA), el device fa un join a la xarxa LoRA, i és en aquest moment en el qual se li assigna un DevAddr, i mitjançant les claus prefixades al dispositiu es generen unes claus de sessió, que canvien cada cop que el device genera una nova sessió.

En l'activació ABP, l'adreça del dispositiu DevADR i les claus de sessió estan preprogramades al device, i resten immutables durant la vida del dispositiu. Aquest mètode simplifica el procés d'unir-se a la xarxa LoRa, ja que no és necessari fer el join, però impedeix que el dispositiu pugui unir-se a una altra xarxa LoRa que aquella predefinida a través del seu DevAddr. Per aquest motiu hem decidit que en la nostra plataforma els devices utilitzin el metode OTAA.

TAULA 1: RELACIÓ SF - BIT RATE - PAYLOAD SIZE

Data Rate	SF	Birate, bps	Payload size
0	12/125	250	51
1	11/125	440	51
2	10/125	980	51
3	9/125	1760	115
4	8/125	3125	242
5	7/125	5470	242
6	7/250	11000	242

La xarxa LoRaWAN és de tipus centralitzada i jeràrquica. De fet, els nodes no poden comunicar-se directament entre ells. Totes les transmissions de dades es produeixen entre els nodes i els gateway, tot i això, els dispositius LoRaWAN permeten una comunicació bidireccional. Depenent de com s'implementin les finestres de recepció, podem classificar els devices en tres classes diferents:

- Classe A: Els missatges d'Uplink (des del dispositiu al servidor) poden ser enviats en qualsevol moment. Quan es realitza un uplink, s'obren dues finestres de recepció (RX1 i RX2) en les quals el dispositiu espera un possible downlink del servidor. Si aquest downlink no arriba, la següent vegada que es pugui rebre informació en aquest dispositiu serà durant les dues finestres del pròxim uplink.

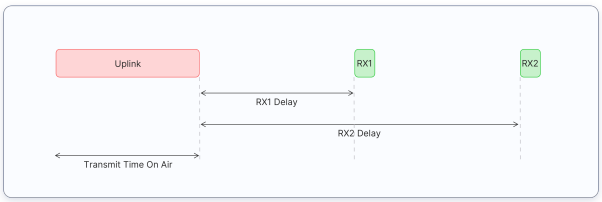


Fig. 5: Dispositiu LoRaWan de Classe A

- Classe B: Els dispositius de classe B permeten rebre missatges downlink sense necessitat d'enviar paquets d'uplink. Això s'aconsegueix mitjançant l'enviament periòdic de beacons per part del gateway. Aquests beacons permeten que els dispositius se sincronitzin amb el gateway, i d'aquesta forma es poden negociar els temps de recepció de paquets des del gateway al dispositiu.

  Aquests dispositius tenen un major consum d'energia que els dispositius de classe A causa del fet que periòdicament s'han de rebre els beacons des del gateway.

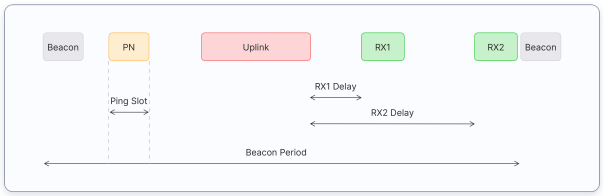


Fig. 6: Dispositiu LoRaWan de Classe B

- Classe C: Els dispositius de classe C estenen les funcionalitats dels de classe A mantenint la finestra de recepció oberta mentre no estiguin transmitent. Això permet d'una comunicació de baixa latència, però a causa d'haver d'estar escoltant contínuament la xarxa de ràdio, el consum energètic és molt més alt.

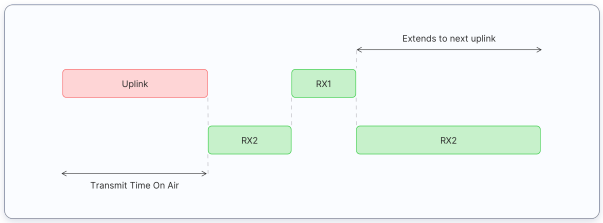


Fig. 7: Dispositiu LoRaWan de Classe C

4.1.2 Estructura de dades del payload

L'estructura de dades utilitzada per tal de transmetre les dades (Payload) entre els devices i la xarxa LoRa és una versió simplificada de l'especificació de Cayenne LPP 2.0[15]. S'ha escollit modificar aquest protocol, per tal de poder simplificar l'enviament de les dades tot mantenint la flexibilitat i baixa utilització de bytes de l'especificació original.

TAULA 2: FORMAT DE DADES DEL PAYLOAD

ID	Type	Data	...	ID	Type	Data
1 byte	1 byte	n bytes	...	1 byte	1 byte	n bytes
51 bytes (SF12) 242bytes max (SF7)						

El protocol de transmissió de dades entre el NS i el AS, tal com s'ha indicat anteriorment, serà HTTP. En els dos proveïdors de xarxa analitzats, els missates rebuts del proveidor a través d'HTTP son en format JSON, pero hi ha diferencies en el nom dels camps i l'ordre en que envien els mateixos, motiu pel qual es necessari de tractar els missatges de forma diferent depenent del proveidor des d'on ens arribin, tal i com s'especifica en l'apartat 4.3.1.

4.2 Gateway/Edge

Cada gateway LoRa està associat amb un determinat Network Server (proveïdor). Entre els proveïdors més importants en l'àmbit mundial podem destacar The Things Network[10], Loriot[12] o Everynet[11]. A nivell mes local, les operadores de telefonia estan apostant per oferir també les seves pròpies solucions. A França, Orange ofereix un servei basat en tecnologia LoRaWAN, que cobreix el 99% del territori frances[13]. A Espanya, l'aliança entre Redexia i Everynet[20] ja cobreix la totalitat de les poblacions de Madrid i Barcelona. A Espanya, igualment, Vodafone i Movistar ofereixen serveis NB-IOT sobre bandes propietàries de les operadores de telefonia.

Considerem que l'Edge del nostre projecte comprèn els gateway LoRa que rebran els paquets dels nostres devices, i els Network Server que rebran els paquets i que els reenviaran al nostre aplication server (AS) situat al nostre cloud.

Per a la realització d'aquest treball hem suposat que hi haurà un gateway amb cobertura disponible d'algun dels proveïdors escollits, en cas contrari, s'haurà d'instal·lar un gateway a la zona per tal de donar-ne.

Dins de les integracions que The Thinks Network té amb els AS destaquen la integració mitjançant HTTP, MQTT i MyDevices[3], mentre que Everynet té integració HTTP i MQTT. En el cas de MQTT, The Things Network té servidor propi en el qual se subscriuen els tòpics, mentre que amb Everynet, el servidor MQTT l'hi ha de posar l'usuari. Per tal de reduir la complexitat de la nostra plataforma i fer-la més portable, he escollit que la integració es faci a través de HTTP.

4.2.1 Creació dels devices al proveidor de xarxa

Cada un dels nostres nodes necesita un Device EUI, Application EUI i App Key per tal de fer el join OTAA. Per al DeviceEUI hem utilitzat el que el propi fabricant assigna al modul LoRa. En el cas de la nostra placa, aquest es troba serigrafiat en una etiqueta situada a sobre del processador[Figura 8].

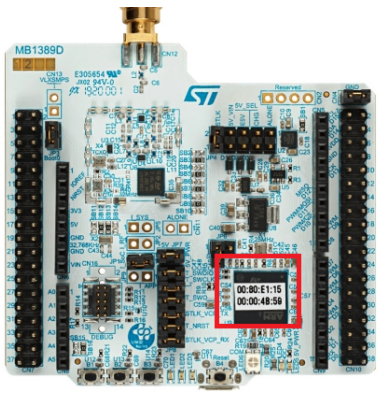


Fig. 8: Etiqueta on es mostra el DevEUI de la nostra placa de desenvolupament

L'application EUI i la APP Key les hem de treure del proveidor de xarxa escollit. A la figura 9 veiem la pantalla per donar d'alta un nou dispositu a TTN, a on es mostra també application EUI i la APP Key del dispositiu que estem creant. A la figura 10 es mostra la mateixa pantalla per part del proveidor d'Everynet.

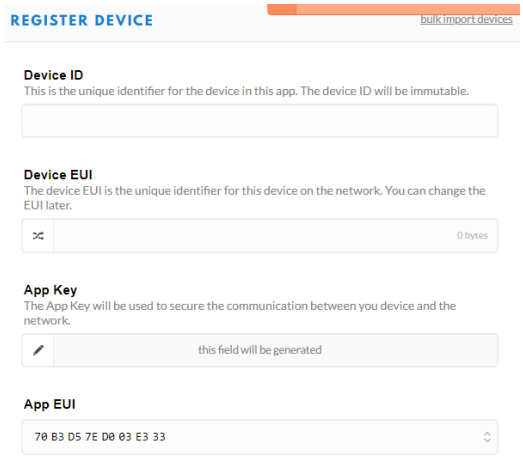


Fig. 9: Registre d'un nou dispositiu a la interficie de TTN

4.3 Cloud

4.3.1 BackEnd

El BackEnd de la nostra plataforma consta d'un servidor HTTP amb suport per a PHP. La funció del BackEnd és rebre les dades des dels NS de The things Network i Everynet, processar-les i emmagatzemar-les a la BBDD final, a més a més, també comprova periòdicament que no s'hagi d'enviar un mail a l'usuari indicant que un del devices ha superat el temps màxim sense que rebem cap dada d'aquest.

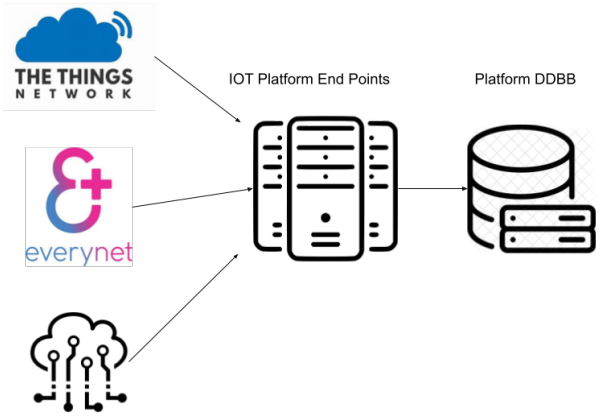


Fig. 11: Arquitectura del backend del sistema

Per tal de poder guardar les dades que es reben, hem implementat un punt d'entrada (en forma d'url única) per a cadascun dels proveïdors de xarxa escollits. Per al cas de The Things Network el punt d'entrada es http://{domain}/endpoints/ttn.php i per al cas d'Everynet http://{domain}/endpoints /everynet/. D'aquesta manera per a poder afegir nous proveïdors de xarxa s'hauria de crear un nou punt d'entrada.

Les dades rebudes per a cada proveïdor són tractades per cada un dels seus punts d'entrada que hem programat. La informació comuna que tots els proveïdors envien és per una banda el payload que el nostre device ha enviat, el DevEUI del device i l'hora en la qual s'ha rebut aquest missatge. Aquestes dades, una vegada tractades, són emmagatzemades a la nostra BBDD. A l'annex 2 es pot veure el disseny de la BBDD que hem creat per a la nostra platafor-

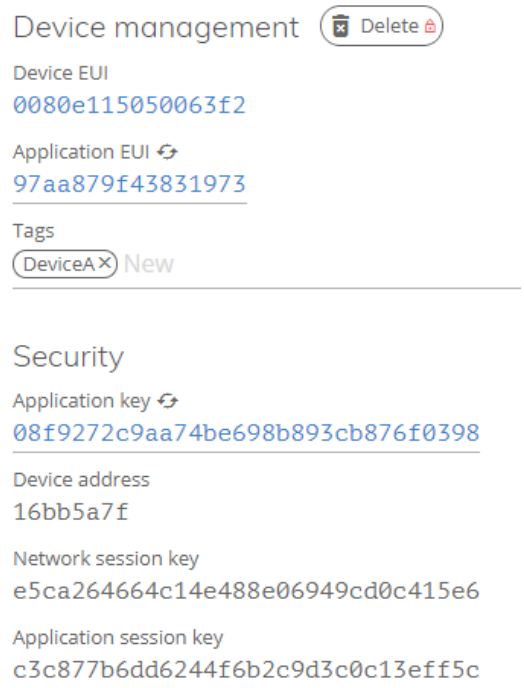


Fig. 10: Registre d'un nou dispositiu a la interficie d'Everynet

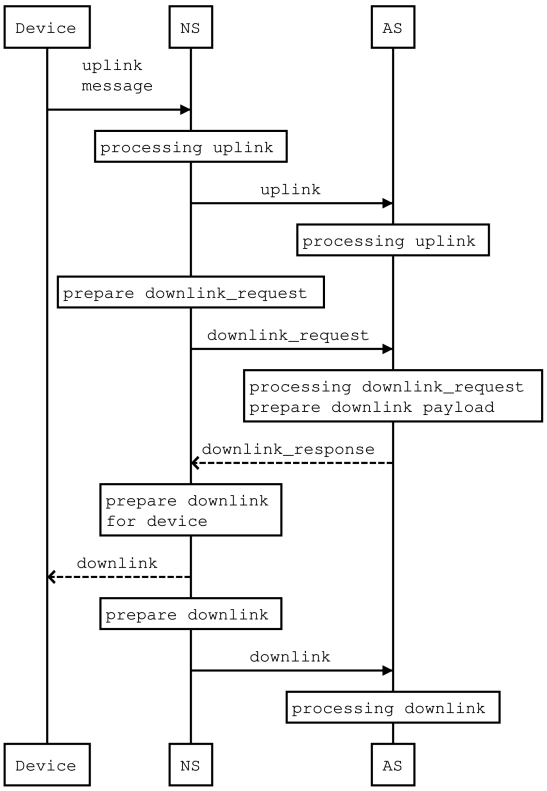


Fig. 12: Workflow d'events per al proveidor Everynet

ma.

En el cas d'Everynet, aquest ens envia periòdicament missatges de tipus OPTIONS als quals hem de respondre amb un 200 OK HTTP. En el cas que no responguem, es desactiva l'enviament de dades [16]. Si responem correctament, rebrem els uplinks a la direcció http://{domain}/endpoints/everynet/uplink i els downlink request a http://{domain}/endpoints/everynet/downlink_request.

Pel que fa als missatges de baixada, o downlink, diferenciem dos casos diferents. Per una banda estan els proveïdors passius, en els quals l'inici d'una petició de missatge de baixada ha de provenir desde la nostra aplicació, i per altra banda, tenim els proveïdors actius, que notifiquen de forma activa a la nostra aplicació quant s'ha obert una finestra de baixada per un dels dispositius.

TTN es un exemple de proveidor passiu, mentres que Everynet es un proveidor actiu. En el cas d'Everynet és el seu NS el que ens notifica (a través d'una crida al nostre downlink_request endpoint) que s'ha obert una finestra per a transmetre dades a algun dels nostres devices. És en aquest moment en què la nostra aplicació pot respondre amb les dades que volem enviar a aquell device, en cas que ho necessitem. A la figura 12 es mostra el workflow de l'enviament i rebuda de dades per a Everynet.

Cal mencionar que encara que per a cada uplink d'Everynet rebrem un downlink request només l'hem de respondre a aquest en cas que efectivament necessitem enviar dades cap al device. En el workflow això es mostra mitjançant una línia puntejada per als downlink response i el downlink del NS al Device.

Per tal d'executar l'enviament dels avisos a l'usuari per mail, i de efectuar els downlink als diferents proveidors, s'ha implementat un script en php, anomenat cron, que ha d'esser executat periodicament, i que es l'encarregat d'enviar els avisos per mail a l'usuari i enviar les peticions de downlink per als proveidors de xarxa de tipus passiu.

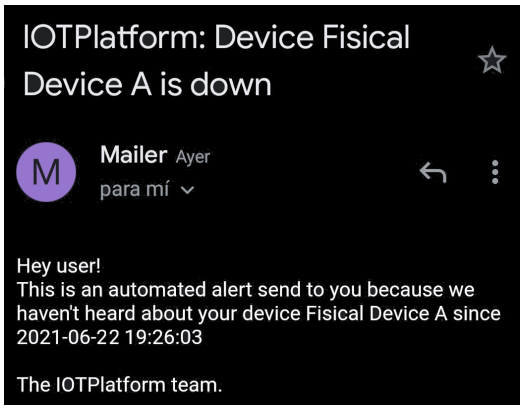


Fig. 13: Exemple de l'email que l'hi arriba a un usuari quan un device perd la connexió.

4.3.2 Frontend

El frontend permet que cada usuari administri els seus propis dispositius, els sensors i actuadors connectats als mateixos i els disparadors, per aquest motiu, cada usuari de la plataforma té un nom d'usuari i contrasenya per tal de poder entrar al sistema. El frontend, quan un usuari s'ha identificat al sistema, consta dels menús My Dashboard, Devices, Sensors i Triggers. El primer que veu l'usuari és el seu quadre de control [Figura 14], on se li presenta de forma gràfica les dades rebudes pels sensors dels seus devices. A l'apartat Devices, se li mostra una taula amb els devices que té

donats d'alta, i se li presenta la possibilitat d'editar-los o eliminar-los.

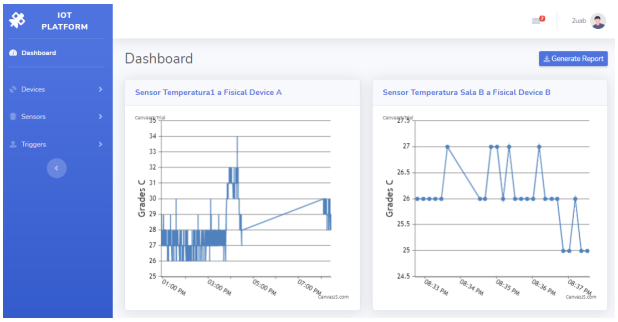


Fig. 14: Dashboard de l'usuari.

L'usuari pot editar el nom i el DevEUI d'un device. Canviar el DevEUI és útil, per exemple, en el cas que un dels devices s'hagi espatllat i s'hagi de substituir, per tal de mantenir les dades rebudes pel device abans d'espatllar-se amb les que arribin amb el nou device. El nom del device permet a l'usuari identificar mitjançant un nom més amigable cada un dels seus device.

Quan un usuari edita un device, també se l'hi mostren els sensors que té associats, i pot editar els mateixos o eliminar-los. Des de la finestra d'edició del device és on l'usuari pot activar que el sistema l'avisi de quan un device fa molt temps que no envia cap dada. El temps en què ha de transcórrer abans d'activar aquesta alerta és configurable de forma individual per a cada device, i es pot desactivar també[Figura 15].

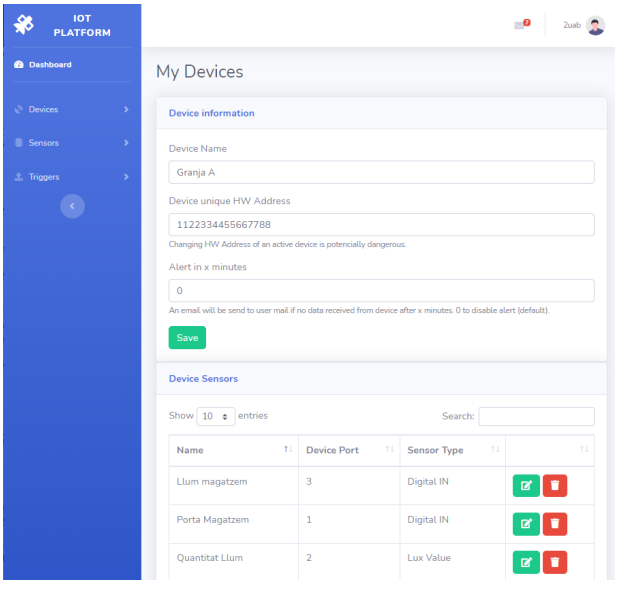


Fig. 15: Al modificar un device, podem esborrar els sensors conectats a ell.

Des del menu sensors l'usuari pot llistar tots els sensors que ha creat, i crear-ne de nous. Per a cada sensor es mostra el tipus de sensor que es (temperatura, humitat, sensor de gas, llum, etc...), així com el device i port al qual esta conectat [Figura 16]per últim, podem veure també les dades rebudes del mateix en forma de grafic [Figura 17].

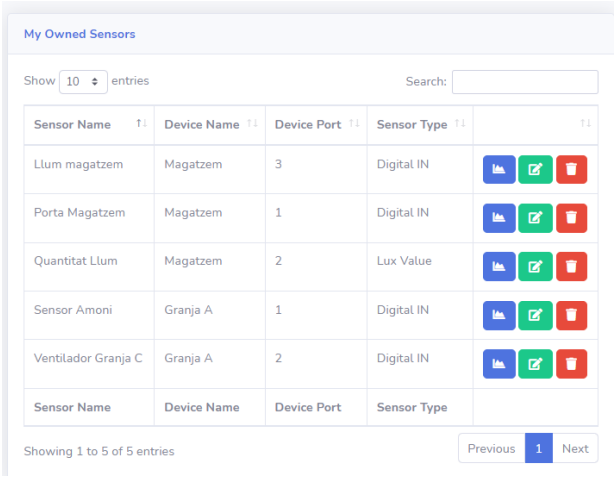


Fig. 16: L'usuari pot modificar o eliminar tots els sensors associats al seu usuari.

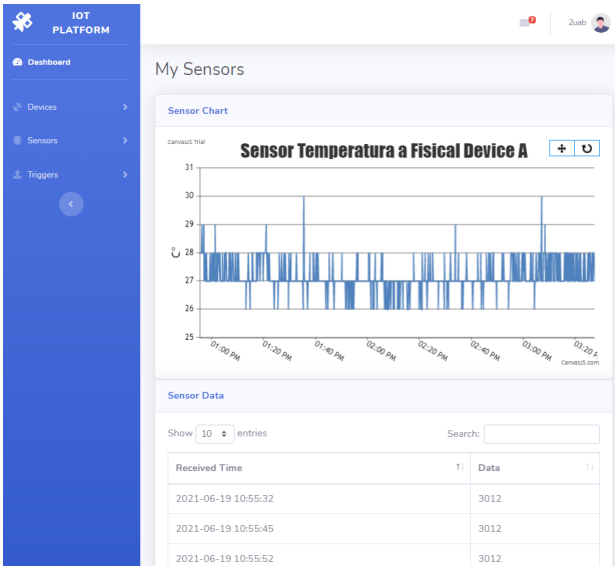


Fig. 17: Les dades d'un sensor es poden mostrar de forma grafica accedint a la seva pagina d'edició.

5 Resultats

El sistema implementa tota la cadena IoT per al NUCLEO-WL55JC, The Things Network i una aplicació de servidor amb back-end i front-end, que permet als usuaris afegir nous sensors i actuadors al sistema. A més, permet avisar l'usuari mitjançant una alerta per email en el cas que un dispositiu no hagi reportat cap dada en un període de temps determinat, la qual cosa permet detectar devices que s'han quedat sense bateria, o detectar que un gateway no funciona correctament, en cas de no rebre cap dada dels devices als quals hauria de donar cobertura.

Així mateix, el front-end desenvolupat permet la visualització interactiva de les dades rebudes dels sensors en forma de gràfics, a través del dashboard o bé accedint al seu gràfic corresponent. Dins d'aquest frontend l'usuari pot configurar disparadors que s'activin depenent de les dades que es rebin dels devices, permetent, per exemple, enviar un senyal a un dels devices per activar un ventilador en cas que la temperatura rebuda per un sensor sigui superior a un valor establert.

La portabilitat de xarxes s'aconsegueix en els devices cambiant el seu APP EUI i APP Key. Per tal de realitzar el canvi s'ha de pujar al device el firmware corresponent al proveidor que estiguem utilitzant per a aquell device.

En el cas del cloud, la portabilitat entre xarxes prové d'haver separat el tractament de les dades rebudes dels proveïdors de la lògica del nostre sistema, mitjançant l'ús d'un adaptador diferent per a cada proveïdor, que és l'encarregat de fer el tractament inicial de les dades rebudes per a transformar-les en un format comú per a tots els proveïdors.

Per tal de validar que les diferents parts del projecte funcionen d'acord amb els objectius finals s'han provat per separat al llarg del projecte.

Per la part dels devices, la manera de comprobar-ho ha estat verificant a la consola del proveidor de xarxa que ens arribaven les dades enviades dels sensors en el format de dades que haviem escollit. Des de la consola del proveidor de xarxa també se'ns permet realitzar un downlink als devices, i d'aquesta manera comprobar que els devices rebien correctament les dades, i eren capaços de descodificarles, per tal de discernir sobre quin actuador voliem actuar.

Per la part del backend, es va simular l'enviament de dades des dels proveïdors de xarxa mitjançant l'eina curl, que ens permet fer crides HTTP des d'un ordinador al nostre servidor i enviar-li d'aquesta manera les dades JSON. A través de la consola del proveidor es va poder comprovar també que la nostra plataforma generaba correctament els missatges de downlink per als devices.

El prototip final creat permet una comunicació bidireccional amb els devices de The Things Network, permetent així que les dades rebudes del sensor d'un device activi l'actuador d'un altre device. Aquesta comunicació no està limitada al mateix proveïdor, d'aquesta manera, és possible que un device que es troba en un dels proveïdors LoRa interactuï amb un altre device amb cobertura d'un altre operador. En aquest aspecte, la integració amb el segon proveïdor de xarxa (Everynet) és únicament teòrica, a causa de les complicacions que ens han sortit de respondre correctament als esdeveniment que espera Everynet, per tant, queda com a treball futur. Tot i això, l'analisis realitzat a 4.3.1 sobre les estructures de dades i les integracions que ens permet aquest proveïdor ens demostra que la seva integració en el sistema és possible.

La quantitat total de codi propi generada és mostra a la taula 3, comprenent un total d'unes 4350 línies. La quantitat total és major, ja que només s'ha tingut en compte el codi generat, i no el codi reutilitzat per llibreries externes. En el cas del device és on es pot observar més diferencia entre codi generat vs. codi total, ja que la major part del codi és generat automàticament per l'IDE.

TAULA 3: QUANTITAT DE LINEES DE CODI GENERADES

Device	150
Backend	700
Frontend	3500

6 CONCLUSIONS

S'ha aconseguit l'objectiu de crear una plataforma de codi obert per a dispositius LoRaWAN, que permet la utilització de diferents proveïdors de xarxa, sense necessitat d'haver de modificar el core de la nostra plataforma, ja que la integració amb els mateixos queda separada en els adaptadors que hem de codificar per a cadascun dels nous proveïdors.

Tenim també un prototip funcional de firmware, que envia correctament les dades als proveidors de xarxa, i aquestes les rebem correctament a la nostra plataforma, a més, per modificar quin proveidor de xarxa utilitzar, l'unic que s'ha de fer es cambiar les claus de xarxa, per tant, es un firmware independent del proveidor de xarxa que utilitzem, que era un altre dels objectius fixats.

Oferim el codi obert que hem generat al públic al github de l'annex, per tal que qualsevol persona pugui realitzar les modificacions que necessitin per adaptar-les a les seves necessitats.

La plataforma que hem creat compleix per tant els objectius fixats, i obre les portes a què es converteixi en un projecte que permeti potenciar l'ús d'aquestes xarxes al gran públic. L'aplicació pràctica de la nostra plataforma compren un gran nombre d'aplicacions, com per exemple sistemes de control de la qualitat d'aire, o aplicacions per a l'agricultura intel·ligent.

7 TREBALL FUTUR

Les línies de futur seria verificar la compatibilitat del nostre sistema, amb la plataforma HW desenvolupada en paral·lel en el TFM d'Enginyeria de Telecomunicacions. En aquesta plataforma HW, implementar en el firmware la lectura de dades mitjançant sensors físics, utilitzant les entrades i sortides específiques, ja que en el nostre treball s'han simulat. Per últim, fer mesures de consum, per tal de verificar que s'utilitzen els modes de baix consum dels que és capaç el xip Arm® Cortex®-M4/M0.

Per altra part, s'hauria de realitzar la integració amb Everynet, codificant la part de la nostra aplicació per respondre correctament a les dades d'aquest proveïdor.

El ventall disponible de disparadors al sistema també té recorregut de millora. Per exemple, la implementació d'un avís per mail quan un sensor sobrepassi un llindar específic, o que l'activació d'un actuador depengui de l'entrada de més d'un sensor, en comptes d'un únic sensor com succeeix en el sistema actual.

Per últim, per tal d'assegurar l'escalabilitat del sistema, s'haurien de fer test de rendiment del backend i frontend. En el backend, per veure com es comporta en rebre centenars de connexions per minut, i el frontend per comprovar el rendiment quan la base de dades té milions de dades guardades.

8 AGRAÏMENTS

El desenvolupament d'aquest treball no hauria estat possible sense el suport ofert en tot moment pels tutors del TFG, Jordi Carrabina i Marc Codina, que en tot moment han estat disponibles per a qualsevol dubte, aportant a més noves idees i proporcionant-me el seu punt de vista sobre el desenvolupament d'aquest. Agrair també a Diego, per haver-me iniciat en l'apassionant món de LoRa, i a Carlos, per la seva ajuda durant el desenvolupament del frontend. Per últim, a

la meva família, sense el suport de la qual no hagués pogut recórrer el camí recorregut durant aquests quatre anys.

REFERÈNCIES

[1] Mariano Pulpito, Paolo Fornarelli, Claudio Pomo, Pietro Boccadoro, Luigi Alfredo Grieco *On fast prototyping LoRaWAN: a cheap and open platform for daily experiments*, (Accedit Maig 2021). [En linea] Disponible: https://ietresearch.onlinelibrary.wiley.com/doi/full/10.1049/iet-wss.2018.5046

[2] Behr Technologies Inc. *6 Leading Types of IoT Wireless Tech and Their Best Use Cases*, (Accedit Març 2021). [En linea] Disponible: https://behrtech.com/blog/6-leading-types-of-iot-wireless-tech-and-their-best-use-cases/

[3] myDevices, *myDevices, the IoT Solutions company*, (Accedit Març 2021). [En linea] Disponible: myDevices https://mydevices.com/

[4] ThingsBoard, Inc, *ThingsBoard Open-source IoT Platform*, (Accedit Març 2021). [En linea] Disponible: https://thingsboard.io/

[5] Thing Big Labs S.L. *Thinger.io – Open Source IoT Platform*, (Accedit Març 2021). [En linea] Disponible: https://thinger.io/

[6] STMicroelectronics, *Building wireless applications with STM32WB Series microcontrollers - Application note*, Capítol 4.4 - Secuencer (Accedit Març 2021). [En linea] Disponible: https://www.st.com/resource/en/application_note/dm00598033-building-wireless-applications-with-stm32wb-series-microcontrollers-stmicroelectronics.pdf

[7] The LoRa Alliance *Homepage - LoRa Alliance®*, (Accedit Març 2021). [En linea] Disponible: https://www.lora-alliance.org/

[8] Ferran Adelantado; Xavier Vilajosana; Pere Tuset-Peiro; Borja Martinez; Joan Melia-Segui *Understanding the Limits of LoRaWAN*, (Accedit Març 2021). [En linea] Disponible: https://ieeexplore.ieee.org/abstract/document/8030482

[9] Juha Petajajarvi; Konstantin Mikhaylov; Antti Roivainen; Tuomo Hanninen; Marko Pettissalo *On the coverage of LPWANs: range evaluation and channel attenuation model for LoRa technology*, (Accedit Març 2021). [En linea] Disponible: https://ieeexplore.ieee.org/abstract/document/7377400

[10] The Things Industries *The Things network*, (Accedit Març 2021). [En linea] Disponible: https://www.thethingsnetwork.org/

[11] Everynet *Everynet Home*, (Accedit Març 2021). [En linea] https://www.everynet.com/

[12] LORIOT AG *LORIOT - The LoRaWAN® Network Server Provider*, (Accedit Març 2021). [En linea] Disponible: https://www.loriot.io/

[13] Orange France*Orange IOT Market- Connectivity*, (Accedit Març 2021). [En linea] Disponible: https://iotmarket.orange.com/connectivity.html

[14] The Things Network Catalunya*Taller de explotación de datos*, (Accedit Març 2021). [En linea] Disponible: https://thethingsnetwork.cat/download/ttncat_taller_explotacion_datos.esp.pdf

[15] My Devices*Cayene LPPP specification*, (Accedit Abril 2021). [En linea] Disponible: https://community.mydevices.com/t/cayenne-lpp-2-0/7510

[16] Arm Limited, *HTTP GitBook — Everynet*, (Accedit Maig 2021). [En linea] Disponible: https://ns.docs.everynet.io/adapters/http2.html

[17] Arm Limited, *Free open source IoT OS and development tools from Arm — Mbed*, (Accedit Febrer 2021). [En linea] Disponible: https://os.mbed.com/

[18] Arm Limited, *STM32WL Support - Mbed OS - Arm Mbed OS support forum*, (Accedit Gener 2021). [En linea] Disponible: https://forums.mbed.com/t/stm32wl-support/11714

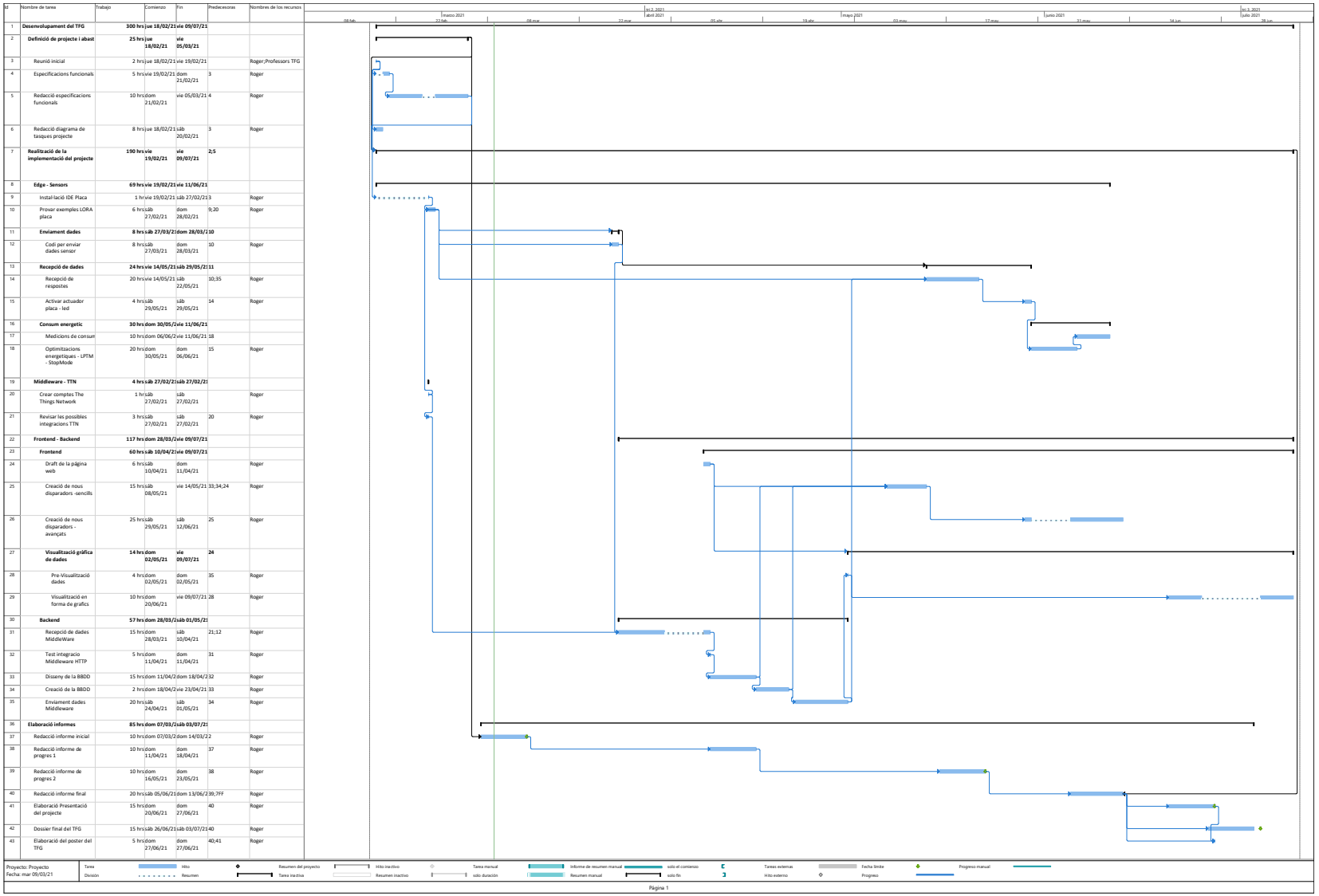
[19] STMicroelectronics, *STM32Cube initialization code generator*, (Accedit Març 2021). [En linea] Disponible: https://www.st.com/en/development-tools/stm32cubemx.html

[20] Everynet, *Spains national IOT network goes live*, (Accedit Abril 2021). [En linea] Disponible: https://www.everynet.com/blog/spains-national-iot-network-goes-live

[21] Danco Davcev, Kosta Mitreski, Stefan Trajkovic, Viktor Nikolovski, Nikola Koteli *IoT agriculture system based on LoRaWAN* , (Accedit Maig 2021). [En linea] Disponible: http://agri.ckcest.cn/file1/M00/02/9C/Csgk0FvuHlmAG4hvAAt6z4aITH0840.pdf

APÈNDIX

A.1 Diagrama de Gantt



Id	Nombre de tarea	Trabajo	Comienzo	Fin	Predecesoras	Nombres de los recursos
1	**Desenvolupament del TFG**	**300 hrs**	**jue 18/02/21**	**vie 09/07/21**		
2	**Definició de projecte i abast**	**25 hrs**	**jue 18/02/21**	**vie 05/03/21**		
3	Reunió inicial	2 hrs	jue 18/02/21	vie 19/02/21		Roger;Professors TFG
4	Especificacions funcionals	5 hrs	vie 19/02/21	dom 21/02/21	3	Roger
5	Redacció especificacions funcionals	10 hrs	dom 21/02/21	vie 05/03/21	4	Roger
6	Redacció diagrama de tasques projecte	8 hrs	jue 18/02/21	sáb 20/02/21	3	Roger
7	**Realització de la implementació del projecte**	**190 hrs**	**vie 19/02/21**	**vie 09/07/21**	2,5	
8	**Edge - Sensors**	**69 hrs**	**vie 19/02/21**	**vie 11/06/21**		
9	Instal·lació IDE Placa	1 hr	vie 19/02/21	sáb 27/02/21	3	Roger
10	Provar exemples LORA placa	6 hrs	sáb 27/02/21	dom 28/02/21	9;20	Roger
11	**Enviament dades**	**8 hrs**	**sáb 27/03/21**	**dom 28/03/21**	10	
12	Codi per enviar dades sensor	8 hrs	sáb 27/03/21	dom 28/03/21	10	Roger
13	**Recepció de dades**	**24 hrs**	**vie 14/05/21**	**sáb 29/05/21**	11	
14	Recepció de respostes	20 hrs	vie 14/05/21	sáb 22/05/21	10;35	Roger
15	Activar actuador placa - led	4 hrs	sáb 29/05/21	sáb 29/05/21	14	Roger
16	**Consum energetic**	**30 hrs**	**dom 30/05/21**	**vie 11/06/21**		
17	Medicions de consum	10 hrs	dom 06/06/21	vie 11/06/21	18	
18	Optimitzacions energetiques - LPTM - StopMode	20 hrs	dom 30/05/21	dom 06/06/21	15	Roger
19	**Middleware - TTN**	**4 hrs**	**sáb 27/02/21**	**sáb 27/02/21**		
20	Crear comptes The Things Network	1 hr	sáb 27/02/21	sáb 27/02/21		Roger
21	Revisar les possibles integracions TTN	3 hrs	sáb 27/02/21	sáb 27/02/21	20	Roger
22	**Frontend - Backend**	**117 hrs**	**dom 28/03/21**	**vie 09/07/21**		
23	**Frontend**	**60 hrs**	**sáb 10/04/21**	**vie 09/07/21**		
24	Draft de la página web	6 hrs	sáb 10/04/21	dom 11/04/21		Roger
25	Creació de nous disparadors -senclls	15 hrs	sáb 08/05/21	vie 14/05/21	33;34;24	Roger
26	Creació de nous disparadors - avançats	25 hrs	sáb 29/05/21	sáb 12/06/21	25	Roger
27	**Visualització gràfica de dades**	**14 hrs**	**dom 02/05/21**	**vie 09/07/21**	24	
28	Pre-Visualització dades	4 hrs	dom 02/05/21	dom 02/05/21	35	Roger
29	Visualització en forma de grafics	10 hrs	dom 20/06/21	vie 09/07/21	28	Roger
30	**Backend**	**57 hrs**	**dom 28/03/21**	**sáb 01/05/21**		
31	Recepció de dades MiddleWare	15 hrs	dom 28/03/21	sáb 10/04/21	21;12	Roger
32	Test integracio Middleware HTTP	5 hrs	dom 11/04/21	dom 11/04/21	31	Roger
33	Disseny de la BBDD	15 hrs	dom 11/04/21	dom 18/04/21	32	Roger
34	Creació de la BBDD	2 hrs	dom 18/04/21	vie 23/04/21	33	Roger
35	Enviament dades Middleware	20 hrs	sáb 24/04/21	sáb 01/05/21	34	Roger
36	**Elaboració informes**	**85 hrs**	**dom 07/03/21**	**sáb 03/07/21**		
37	Redacció informe inicial	10 hrs	dom 07/03/21	dom 14/03/21	2	Roger
38	Redacció informe de progres 1	10 hrs	dom 11/04/21	dom 18/04/21	37	Roger
39	Redacció informe de progres 2	10 hrs	dom 16/05/21	dom 23/05/21	38	Roger
40	Redacció informe final	20 hrs	sáb 05/06/21	dom 13/06/21	39;7FF	Roger
41	Elaboració Presentació del projecte	15 hrs	dom 20/06/21	dom 27/06/21	40	Roger
42	Dossier final del TFG	15 hrs	sáb 26/06/21	sáb 03/07/21	40	Roger
43	Elaboració del poster del TFG	5 hrs	dom 27/06/21	dom 27/06/21	40;41	Roger

A.2 Disseny BBDD

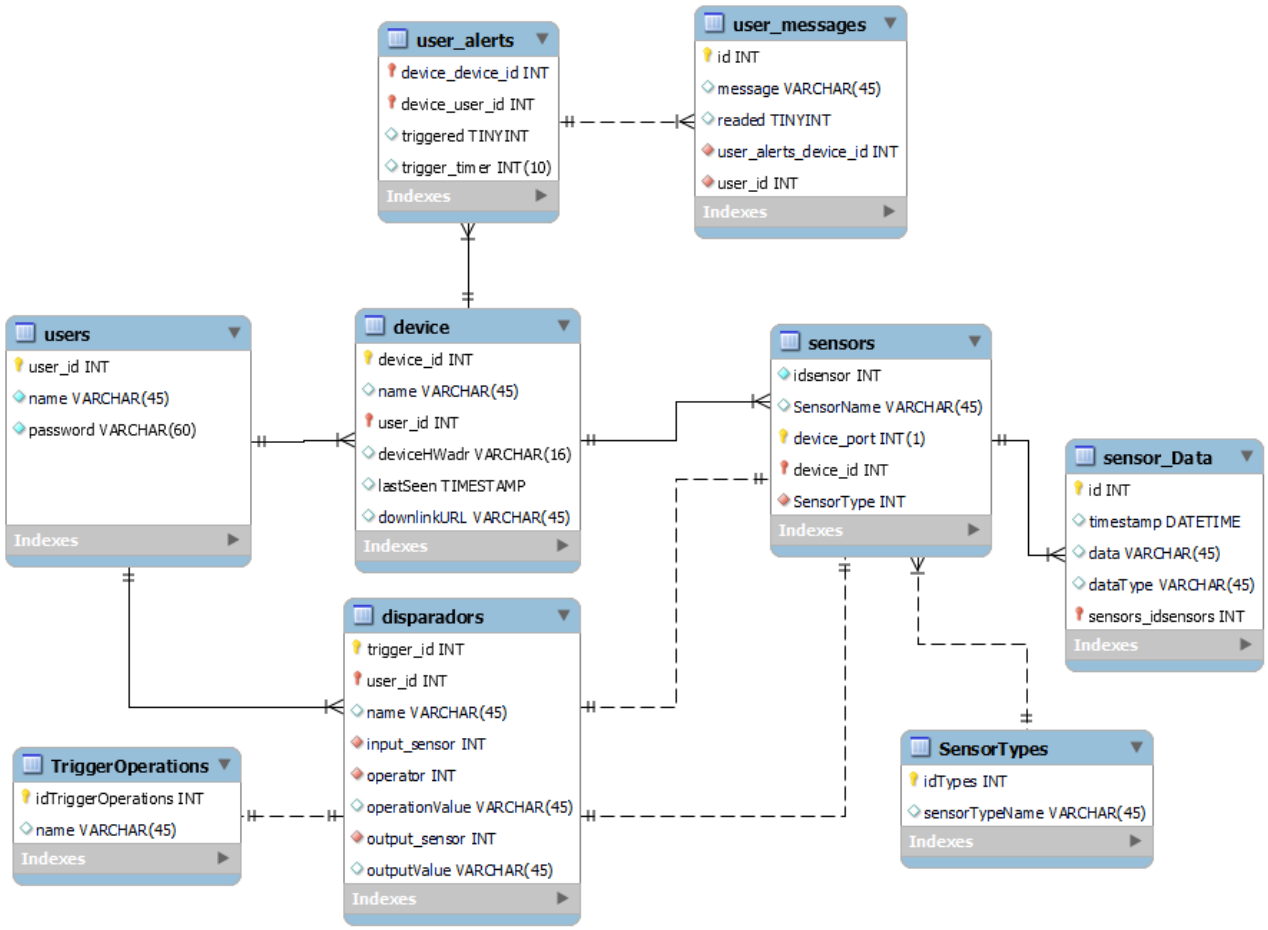


Fig. 18: Estructura de la base de dades

A.3 Codi del projecte

El codi del projecte es troba disponible als seguents repositoris:
Codi per als Sensors
https://github.com/Wewannado/TFG-LoraSensor
Codi del Fronted End i BackEnd desenvolupat
https://github.com/Wewannado/TFG-LoraPlatform